Detecting Duplicates In Public Network Using Signatures

J.Sravanthi#1, G.Kalyani#2

#1 Student, Dvr & Dr. Hs Mic College Of Technology, Kanchikacherla,Krishna(dt)

#2 Assoc. professor, Dvr & Dr. Hs Mic College Of Technology, Kanchikacherla,Krishna(dt)

#1 sravanthij.888@gmail.com, #2 kalyanichandrak@gmail.com,

Abstract: Data Leakage is a silent type of threat. For example, an employee can intentionally or accidentally leak certain sensitive information. Sensitive information may be any data which an organization considers to be privileged and confidential. Hence which sensitive data that has already been leaked from the enterprise and is publicly available, for example, on the Internet should be detected. This strategy is post-facto leakage detection. Traditionally, this leakage detection is handled by watermarking, in which a unique code is embedded in each distributed copy. By introducing a technique beyond watermarking, we can facilitate this post-facto detection technique, in which a unique embedded signature will be identified from within the contents of the original document containing the sensitive data. In this paper, We present an automated tamper-proof low complexity algorithm to solve data leakages. We extract *embedded signatures* from sensitive documents and use them in conjunction with search engines to determine whether near-duplicate versions of the document (or portions of it) are available on the Web. The embedded signature is tamper-proof; even if an adversary partially modifies a document, our mechanism can detect duplicate copies. Also, if a duplicate copy is present in the Web, our system can detect such a copy with a small number of queries.

I INTRODUCTION

Sensitive information within an enterprise can be categorized into two kinds: private data not accessible to the external world and copyrighted information that is made publicly available. Information leakage can occur due to a wide range of factors and not all are preventable using technical means. Though enterprises have invested in a wide-range of leakage prevention solutions, none of these mechanisms are fool-proof.

Data outflow analysis to detect leakage is used in two different Ways: a) to prevent leakage and b) to detect leakage after it has occurred ("post-facto"). An important goal of data leakage prevention is to develop a mechanism that will prevent any unauthorized user or process from improperly "leaking" any one of a given set of pre-identified Sensitive documents. An important goal of post-facto leakage detection is to develop a mechanism that will determine which sensitive data has already leaked from the enterprise and is publicly available, for example, on the lntemet. A common method for facilitating post-facto leakage detection is to use Watermarking. Watermarking generally involves modifying a document in some Way to make the document more distinguishable than it was before the Water-marking. These modifications may either be visible or invisible to an observer. The Watermark is

then used to detect a document that has been improperly leaked.

While Watermarking does help to distinguish a document, the technique has several Weaknesses. First, since Watermarking involves adding something to a document, this technique requires recognizing, before a leakage occurs, that a document needs to be watermarked. Second, watermarking is subject to tampering. A malicious party who seeks to make pirated information indistinguishable may be able to remove an added Watermark. Accordingly, an improved method beyond traditional Watermarking strategies is needed for detecting the post-facto leakage of sensitive information into a public domain, such as the lntemet. The method should be tamper-resistant, meaning that the sensitive electronic document should remain detectable even if it has been partially modified.

In this paper, we focus on the problem of information leakage detection on the Web. Given that enterprises typically own volumes of sensitive data, our goal is to determine in an efficient manner if a small fraction of the sensitive documents have leaked on the Web. We describe an automated mechanism to address problems. A key idea in our detection algorithm is to extract an *embedded signature* from a document that can be used to achieve the following properties: First, one can construct appropriate search queries from an embedded signature that can detect near-duplicate versions of documents (or portions of a document) using a small number of queries. Second, the embedded signature construction ensures that the signature is resilient against tampering; adversarial partial tampering of the document cannot evade detection. Third, we ensure that every term in an embedded signature is a commonly occurring term and thereby does not leak any sensitive information to the search engine as part of a query.

II RELATED WORK

Recent work on finding near-duplicate pages using crawling the Web uses the earlier stated random projection based fingerprinting method to generate unique low dimensional fingerprints for each Web document. The unique fingerprint for the document is compared with other documents on the Web to find a match. While this fingerprint based method provides a compact metric for measuring similarities across documents, this idea cannot be directly used in our context due to two reasons. First, the fingerprinting based approach is not tamper-proof. Second, the nature of the fingerprints (based on a 64-bit hash) generated for each page is not conducive to construct appropriate search queries. The idea of finding near-duplicate documents within a repository has been used in various contexts including electronic copyright system, Web clustering, filtering duplicated messages from the newsgroup/mailing-list, finding similar file in a large file system, checking plagiarism in student exercises etc.

In signature-based schemes the document is traced with the help of a signature explicitly embedded in the document, such as adding a watermark. The key idea behind such mechanisms is to modify the document by embedding certain codeword's into the document in such a way that they are not detected by human eye but can be detected by the computer. The most common document coding methods are line-shift coding, word-shift coding and feature-coding. Where the documents are altered by vertically (horizontally) shifting the location of lines (words), or certain alphabet shapes are changed at certain positions. The white spaces at the end of the lines can also be used to hide information. The problem with such mechanisms is that the watermark can be *removed* automatically and the document can no longer be identified. Moreover, the watermarking techniques require modification to the original documents. Another approach to embed signatures within documents is to use natural language processing techniques, where the document is modified without changing its meaning. These techniques effect the lyrical quality of the document and cannot be applied to certain types of text documents.

Copy-prevention mechanisms include physical isolation of information which uses a special purpose hardware for authorization, and active documents which are documents produced by programs. The problem with these mechanisms is that they are restrictive in nature, cumbersome and incomplete and can be defeated. In addition, the issue of redistribution and plagiarism of partial content is not addressed in these works. Many software vendors don't prefer such an approach because they come in the way of users. To achieve strong leakage prevention guarantees on a per-host level, enterprises can potentially leverage recent works on the design of secure operating systems such as HiStar and Asbestos; however, it is unclear whether these systems would be adopted by enterprises.

III PROBLEM DEFINITION

Data leakage problem is motivated by two common information leakage threats facing enterprises: leakage of internal private pages and plagiarism of copyright information on the Web. We want to detect when an adversary leaks a large volume of sensitive data with or without tampering the content. We use a simple threshold-based tampering adversarial strategy. Let $\rho(D)$ denote the fraction of terms that an adversary would tamper in a given document D before publishing it. We use a threshold $\rho 0$ such that $\rho(D) < \rho 0$ for all leaked documents. The aim of our detection algorithm is to trace the copied documents if they are not tampered more than $\rho 0$.

Given a document A which can be represented as a sequence of k terms, we define a document A' to be a *near-duplicate* version of a document A if A' can be derived by $\rho 0 \times k$ modification operations to A. A single modification operation can be one of two changes to the document: (a) a specific term is replaced by another term or a sequence of terms; (b) a sequence of terms is inserted between two terms. To simplify notation, we use the same threshold $\rho 0$ in the threshold based adversarial strategy. Data leakage problem is similar to the problem of finding near duplicate pages on the Web. The key difference is that prior work on duplicate detection relies on a local repository of the Web. Here, we rely on search engines to look for near duplicates. We need to provide appropriate search queries, download some of the results and then check for near duplicates in the downloaded pages.

IV OBJECTIVES

Low-overhead: For a single random document, the overhead of determining near-duplicate copies should be small.

Fast identification: In the event of a bulk leakage, the time to identify the potential suspect domains should be very small and should be independent of the total number of documents.

Tamper proof: Even if each of the leaked sensitive documents are tampered (either randomly or manually) we should still be able to identify the near-duplicate documents.

Reduce self-leakage: Any algorithm that we use to search for near-duplicate copies of a sensitive document, should only leverage commonly occurring terms and not expose sensitive information from the document.

V PROPOSED SOLUTION

Leakage detection would be simplified if an enterprise maintained a local copy of the entireWeb; this is infeasible for most enterprises. Using untrusted third parties that may have a copy of the Web to determine leakage requires sharing sensitive files with the third party. We determine a compact embedded signature by carefully extracting a random collection of terms that can aid in searching for near-duplicate versions of the document on the Web.

Embedded signature construction mechanism is motivated by the idea of contiguous subsequence contained in *D*. Our approach uses the idea of determining randomized rare contiguous subsequence within a document. A rare contiguous

subsequence is one that has a low frequency of occurrence on the Web and a randomized rare contiguous subsequence is one that appears at a random position within the document. The embedded signature we construct for a document is a collection of a constant number of randomized rare contiguous subsequence for a document.

Embedded signature mechanism satisfies four main properties. First, rare contiguous subsequence form appropriate candidates to search for near-duplicate copies of a document on the Web. Second, we show that, given a random position within the document, one can extract a rare contiguous subsequence from that position. Third, the embedded signature extracted contains a constant number of contiguous subsequence. Finally, to reduce self leakage, we ensure that each chosen contiguous subsequence is not specific to an enterprise and comprises of only commonly occurring terms on the Web.

The key requirement for our SDL algorithm is to determine rare contiguous subsequences in a document at any randomly chosen position. We use the contiguous subsequence frequency database of the Linguistic Data Consortium (LDC) to determine the rarity of a contiguous subsequence in D. LDC has snapshots of the frequency of various contiguous subsequences in D on the Web as recorded by a search engine.

The key steps in the signature generation algorithm are as follows:

SignatureGen(D,K, T,M,GLDC)
D = Document, T = rarity threshold
K= Secret one-way hash function
M= Number of contiguous subsequence in embedded signature
GLDC= Generic LDC contiguous subsequence frequency database
1) Generate a set of M random locations P = {p1,p2,. . . ,pM} in D, using a secret function K.
2) Beginning at position pi in D, determine the smallest shingle wi such that GLDC(wi) < T .
3) Return (Sign (D) = {w1, w2,. . . ,wM}) as the embedded signature of document D.

Given Sign (D), the embedded signature of a document D, the algorithm for searching for near-duplicate versions of document involves the following steps:

Search (Sign (D),N)
Sign(D) = Embedded signature of document D
N= Number of contiguous subsequence in D in a query
1) Randomly choose N different contiguous subsequence q1, . . . qN from the Sign (D)
2) Query Q = q1+q2+...+qN
3) Search (Q) = Search result for query Q from a search engine
4) S = |Search (Q)|,
5) δ (Q) = {list of suspect domains (Search(Q))}
6) Return(S, δ(Q))

The goal of the signature generation and the search algorithms are two-fold: (a) minimize the number of search results, S in a search query to find near-duplicate replicas for a document; (b) be tamper-proof up to a certain threshold ρ0. Signature generation algorithm uses a secret function K to generate random positions to extract rare contiguous subsequence in D. Given that K is unknown to the adversary, the adversary cannot guess the random positions where the contiguous subsequence in D are extracted from. The embedded signature is a collection of M rare contiguous subsequence in D which are randomly chosen from various locations.

We use the embedded signatures to form search queries. A search query is a combinational query comprising of N out of M randomized rare contiguous subsequence in D in the signature Sign(D). Given a query, let S denote the number of pages output by a search engine that we need to download to check for near-replicas. The search results are the pages which have all N rare-contiguous subsequence in D, thereby determining potential copies of the document on the Web. The goal here is to create a search query which minimizes

S. When an adversary has tampered a leaked document, we require multiple queries to search for the modified document. The algorithm can also be applied to portions of a document (as opposed to the entire document) to detect if a specific portion of a document has been leaked.

VI CONCLUSION

Data Leakage is a silent type of threat. Traditionally, this leakage detection is handled by watermarking, in which a unique code is embedded in each distributed copy but it has several weaknesses. In this paper, we focus on the problem of information leakage detection on the Web. Given that enterprises typically own volumes of sensitive data, our goal is to determine in an efficient manner if a small fraction of the sensitive documents have leaked on the Web. We describe an automated mechanism to address problems. A key idea in our detection algorithm is to extract an *embedded signature* from a document that can be used to achieve the following properties: First, one can construct appropriate search queries from an embedded signature that can detect near-duplicate versions of documents (or portions of a document) using a small number of queries. Second, the embedded signature construction ensures that the signature is resilient against tampering; adversarial partial tampering of the document cannot evade detection. Third, we ensure that every term in an embedded signature is a commonly occurring term and thereby does not leak any sensitive information to the search engine as part of a query. The embedded signature is tamper-proof, fast identification, lower overhead; even if an adversary partially modifies a document, our mechanism can detect duplicate copies.

VII REFERENCES

[1] M. Henzinger, "Finding near-duplicate web pages: a large-scale evaluation of algorithms," in *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. New York, NY, USA: ACM, 2006, pp. 284–291.

[2] M. Atallah, V. Raskin, C. Hempelmann, M. Karahan, R. Sion, U. Topkara, and K. Triezenberg, "Natural Language Watermarking and Tamperproofing," *Information hiding*, vol. 2578/2003, pp. 196–212, 2002.

[3] N. Shivakumar and H. Garcia-Molina, "Building a scalable and accurate copy detection mechanism," in *DL '96: Proceedings of the first ACM international conference on Digital libraries*. New York, NY, USA: ACM, 1996, pp. 160–168.

[4] S. Brin, J. Davis, H. Garcia-Molina.*Copy Detection Mechanisms for Digital Documents.* Proceedings of the ACM SIGMOD Annual Conference, San Francisco, CA, May 1995.

[5] A. Z. Broder, "Identifying and filtering near-duplicate documents," in *CPM '00: Proceedings of the 11th Annual Symposium on Combinatorial Pattern Matching*. London, UK: Springer-Verlag, 2000, pp. 1–10.

[6] S. Low, N. Maxemchuk, J. Brassil, and L. O'Gorman, "Document marking and identification using both line and word shifting," *Proceedings of the Fourteenth Annual Joint Conference of the IEEE Computer and Communication Societies (Vol. 2)-Volume-Volume 2*, 1995.

[7] J. Brassil, S. Low, N. Maxemchuk, and L. OGorman, "Electronic Marking and Identification Techniques to Discourage Document Copying," *IEEE JSAC*, vol. 13, no. 8, p. 1495, 1995.

[8] A. Broder, S. Glassman, M. Manasse, and G. Zweig, "Syntactic clustering of the Web," *WWW 1997*, pp. 1157–1166, 1997.

[9] J. Ryder Sr and S. Smith, "Self-verifying receipt and acceptance system for electronically delivered data objects," Aug. 28 1990, uS Patent 4,953,209.

[10] U. Topkara, M. Topkara, and M. Atallah, "The hiding virtues of ambiguity: quantifiably resilient watermarking of natural language text through synonym substitutions," *International Multimedia Conference*, pp. 164–174, 2006.